IN THE CLAIMS:

Please amend Claims 1, 5, 11, 20, and 23. Also, please delete Claims 10, 12 and 27 without prejudice.

1. (Currently amended): A method for video graphics processing, comprising:

receiving primitive vertex parameters corresponding to vertices of a video graphics primitive, wherein the primitive vertex parameters for each vertex include a primitive vertex set of three-dimensional coordinates and a primitive vertex normal vector;

tessellating the video graphics primitive to produce a plurality of component primitives, wherein each component primitive of the plurality of component primitives is defined by component vertices having corresponding component vertex parameters, wherein component vertex parameters for each component vertex include a component vertex set of three-dimensional coordinates and a component vertex normal vector, wherein the component vertex parameters for each component vertex are derived from at least a portion of the primitive vertex parameters;

for each component primitive of the plurality of component primitives:

processing the component primitive using a three-dimensional processing pipeline, wherein processing generates pixel data corresponding to the component primitive; and

wherein tessellating further comprises tessellating based on a tessellation level, wherein the tessellation level determines a number of component primitives included in the plurality of component primitives.

2. (Original): The method of claim 1, wherein tessellating the video graphics primitive further comprises calculating the component vertex parameters for each of the component vertices using linear interpolation.

3. (Original): The method of claim 1, wherein tessellating the video graphics primitive further comprises calculating the component vertex parameters for each of the component vertices using Nth order interpolation, wherein N is an integer greater than one.

4. (Original): The method of claim 1, wherein tessellating the video graphics primitive further comprises re-normalizing the component vertex normal vector included in the component vertex parameters for each of the component vertices.

5. (Currently amended): The [method of 1] method of Claim 1, wherein processing the component primitive includes adding lighting effects to the pixel data based on the component vertex normals for the vertices of the component primitives.

6. (Original): The method of claim 5, wherein adding lighting effects further comprises adding specular lighting effects.

7. (Original): The method of claim 5, wherein adding lighting effects further comprises adding diffuse lighting effects.

8. (Original): The method of claim 5, wherein adding lighting effects further comprises adding environment mapping lighting effects.

9. (Original): The method of claim 5, wherein adding lighting effects further comprises:

calculating vertex lighting effects at each of the component vertices of the component primitive; and

calculating lighting effects for each pixel location in the component primitive by linearly interpolating the vertex lighting effects for at least a portion of the component vertices of the component primitive.

10. (Delete)

10.

N. (Currently amended): A video graphics circuit, comprising:

a frame buffer that stores pixel data corresponding to image data for a frame;

a central processor that generates processing commands and vertex parameters corresponding to video graphics primitives;

a control processor operably coupled to the central processor, wherein the control processor receives the processing commands from the central processor, wherein the control processor generates control information based on the processing commands;

a tessellation block operably coupled to the central processor and the control processor, wherein the tessellation block receives a first portion of the control information from the control processor, wherein, for each video graphics primitive, the tessellation block receives vertex parameters corresponding to each of the vertices of the video graphics primitive and tessellates the selected video graphics primitive based on tessellation information included in the first portion of the control information, wherein tessellation of the video graphics primitive produces component vertex parameters for a plurality of component primitives that correspond to the video graphics primitive;

a lighting block operably coupled to the tessellation block and the control processor, wherein the lighting block receives component vertex parameters corresponding to each of the component primitives from the tessellation block and receives a second portion of the control information from the control processor, wherein the lighting block adds lighting effects to the component vertex parameters for each of the component primitives to produce modified vertex parameters, wherein the lighting effects are added based on at least a portion of the component vertex parameters and the second portion of the control information;

a three-dimensional video graphics pipeline operably coupled to the lighting block, the control processor, and the frame buffer, wherein the three-dimensional video graphics pipeline receives the modified vertex parameters for each of the component primitives and processes each of the component primitives to generate pixel fragment data that is blended with the pixel data stored in the frame buffer; and

wherein the tessellation information includes a tessellation level, wherein the tessellation level determines a number of component primitives included in the plurality of component primitives for each of the video graphics primitives.

12. (Delete)

13. (Original): The video graphics circuit of claim 11, wherein the vertex parameters for each video graphics primitive and the component vertex parameters for each component primitive of the plurality of component primitives include, for each vertex, three-dimensional coordinates and a normal vector.

14. (Original): The video graphics circuit of claim 13, wherein the tessellation block re-normalizes the normal vector included in the component vertex parameters for each component primitive.

15. (Original): The video graphics circuit of claim 13, wherein the tessellation block tessellates each video graphics primitive by calculating the component vertex parameters for each component primitive from the vertex parameters for a corresponding video graphics primitive using linear interpolation.

16. (Original): The video graphics circuit of claim 13, wherein the tessellation block tessellates each video graphics primitive by deriving the component vertex parameters for

each component primitive from the vertex parameters for a corresponding video graphics primitive using N-th order interpolation, wherein N is an integer greater than one.

15. 11
~~17.~~ (Original): The video graphics circuit of claim ~~13~~, wherein the lighting effects added by the lighting block include specular lighting effects.

16. 15
~~18.~~ (Original): The video graphics circuit of claim ~~11~~, wherein the lighting effects added by the lighting block include diffuse lighting effects.

17. 15
~~19.~~ (Original): The video graphics circuit of claim ~~11~~, wherein the lighting effects added by the lighting block include environmental mapping lighting effects.

18.
~~20.~~ (Currently amended): A method for video graphics processing, comprising:

receiving vertex parameters corresponding to vertices of a video graphics primitive, wherein the vertex parameters for each vertex includes three-dimensional coordinates and a normal vector;

tessellating the video graphics primitive to produce a plurality of component primitives, wherein the plurality of component primitives are defined by a plurality of additional vertices and the vertices of the video graphics primitive; and

calculating an additional normal vector for each additional vertex of the plurality of additional vertices; and

wherein tessellating further comprises tessellating based on a tessellation level, wherein the tessellation level determines a number of component primitives included in the plurality of component primitives.

19 18
~~21.~~ (Original): The method of claim ~~20~~, wherein calculating an additional normal vector further comprises re-normalizing the additional normal vector.

~~22.~~ 20. (Original): The method of claim ~~20~~ 18, wherein calculating further comprises deriving each additional normal vector from at least a portion of the normal vectors for the vertices of the video graphics primitive using linear interpolation.

~~23.~~ 21. (Currently amended): The method of claim ~~20~~ 18, wherein calculating further comprises deriving each additional normal vector from at least a portion of the normal vectors for the vertices of the video graphics primitive using N-th order interpolation, wherein N is an integer [great] greater than one.

~~24.~~ 22. The method of claim 20 further comprises processing the plurality of component primitives using a three-dimensional pipeline, wherein processing the plurality of component primitive includes producing pixel data corresponding to the plurality of component primitives, wherein processing the plurality of component primitives further includes adding lighting effects to the pixel data based on the normal vectors and additional normal vectors corresponding to the vertices of the plurality of component primitives.

~~25.~~ 23. (Original): The method of claim ~~24~~ 22, wherein adding lighting effects further comprises adding at least one of specular lighting effects, diffuse lighting effects, and environment mapping lighting effects.

~~26.~~ 24. (Original): The method of claim ~~24~~ 22, wherein adding lighting effects further comprises:

calculating vertex lighting effects at each vertex and each additional vertex corresponding to the plurality of component primitives; and

calculating lighting effects for each pixel location in the plurality of component primitives by linearly interpolating the vertex lighting effects corresponding to a component primitive of the plurality of component primitive in which the pixel location is included.

27. (Delete)